The BinSys application on the EDGeS infrastructure

Attila Csaba Marosi[1], Attila Kovács[2], Péter Burcsi[2], Ádám Kornafeld[1]

[1] MTA SZTAKI, Kende u. 13-17,
Budapest, H-1111 Hungary,
{atisu, kadam}@sztaki.hu

[2] Eötvös Loránd University Budapest,
Department of Computer Algebra,
Pázmány Péter sétány 1/c,
H-1117 Hungary
{attila.kovacs, peter.burcsi}
@compalg.inf.elte.hu

Introduction

The usual decimal system can be generalized in several ways, GNS (generalized number systems) being one of them. In these systems the decimal base 10 is replaced by a square matrix and the digits 0-9 are replaced by a suitable set of vectors, also called digits. In some cases, every vector in the space can be represented by a finite sum of powers of the base each multiplied by one of the digits, the same way as for example 256 is the sum of 100 times 2, 10 times 5 and 1 times 6. In these cases the vectors of the space can be mapped to finite sequences of digits and vice versa - suggesting potential applications in coding or cryptography. These applications explain why these systems are of interest for informatics.

Unfortunately, the exact conditions on the base and the vector that ensure that such a representation exists and is unique are currently unknown, making the problem also interesting for mathematicians. The aim of the application is to provide a complete list of GNS in a few special cases. Although there are an infinite number of systems, if we fix the size of the problem, that is the size and number of the vectors and impose an additional condition of using the so-called canonical digit set, then this number is finite and computer methods for traversing the search space exist. Obtaining the complete lists for the special cases will hopefully support the better mathematical understanding of the structure of GNS. Some partial results are already obtained in [1].

Generalized Number Systems

The BinSys project is concerned with generalized number systems (GNS), one of several multi-dimensional generalizations of number representations. In the traditional decimal number system, numbers can be uniquely represented by a finite string of digits. Each digit stands for a product whose value is an appropriate power of 10 multiplied by the digit, e.g. 256 = 200 + 50 + 6, the first summand being 100 times the first digit, etc. Now if we replace the base 10 by an n-by-n integer matrix M, and we replace the possible digits by n-dimensional integer vectors, we can try to represent integer vectors by a string of digits where the actual value of the digit is a power of the matrix multiplied by the digit vector. We say that the matrix together with the digit set forms a GNS if every integer vector has a unique representation of the required form. In the special case of dimension 1, the matrix as well as the digits is simply numbers. Note, however that the decimal system falls short of being a GNS, since negative numbers can only be represented using a sign: we return to this below. A 1 dimensional example that works can be obtained using base -10 for example. It is not hard to show that the following conditions are necessary for a GNS:

1. The matrix has to be expansive which means that the eigenvalues of the matrix are larger than 1 in absolute value,
2. The digit vectors have to form a complete residue system modulo the matrix which is equivalent to saying that there are $|\det M|$ of them, and that the difference of neither two distinct digits is of the form Mv with an integer vector v.

Note that for the decimal system, the first condition simply says that $10 > 1$, while the second states that there are 10 digits, pair wise incongruent modulo 10. Unfortunately, exact conditions on the base and the vectors that characterize the GNS property are unknown. The project's main scope is providing a complete list of GNS in a few special cases. Although there are an infinite number of systems, if we fix the dimension, the determinant and the form (rational normal form or companion matrix form) of M and only allow so-called canonical digit sets, then this number is finite. Even these special cases are not fully understood so a computer-aided search could provide some insight. The first part of the search is a computationally intensive search for expansive polynomials (the characteristic polynomials of the candidate matrices) which produces a complete list that can then be checked for the GNS property reasonably fast on a single desktop machine.
Expansivity of polynomials can be checked without actually computing the roots, using the Lehmer-Schur algorithm. This algorithm also gives bounds on the coefficients of expansive polynomials as a function of the other coefficients. This gives rise to a natural parallelization: fix some coefficients in several different ways, letting the search for all possible values of the unfixed coefficients be one of the distributed chunks. Each chunk can be performed separately from the others, the only communication takes place when the found expansive polynomials are reported. If the chunks are still too large, they can be further subdivided using a special ordering on the polynomials.

BinSys Application

BinSys consist of three parts: i.) an input generator; ii.) a master application; and iii.) a client application.

The program that generates the distributed chunks is called input generator. It has the following parameters: the degree and the constant term of the polynomial (they correspond to the dimension and the determinant of the base matrix), the leading coefficient (for future use, currently it is 1), and a parameter called depth that determines the number of fixed coefficients per chunk. The output of the program is a sequence of lines where each line specifies a portion of the search space. A line contains a line number, information about the degree and the first unfixed coefficient and two polynomials, indicating the start and the end of the search space corresponding to the line.

This generated file is quite large and holds many lines depending on the size of each chunk, e.g. 2.9GB and 24M lines total for a typical 13 dimension input generation.

The master application is responsible for creating tasks from the input file, thus copying lines (portions of the search space) from the input file into a separate file that serves as input for a given task. Since the processing-time of a line is not constant even within the same dimension and chunk size, and a single task may consist of more than one line, predicting the exact processing time for a line is not possible, but estimates can be given.

The client application performs the actual search (detailed in the previous section) in a given portion of the search space, represented by a line of text in the input file. Since tasks consist of more lines, the most effective way of implementing check-pointing was to do it every time after search finishes for a given line in a task. By adjusting the "size" of the line (thus the portion of the search space it covers), and the number of lines within a task, the estimated computation time required for processing a line (thus the time elapsing between two check-points) becomes "optimal" in terms of reducing the check-pointing interval.

Since the application was originally implemented using DC-API thus it is fully compatible with EDGeS Bridge Services [2].

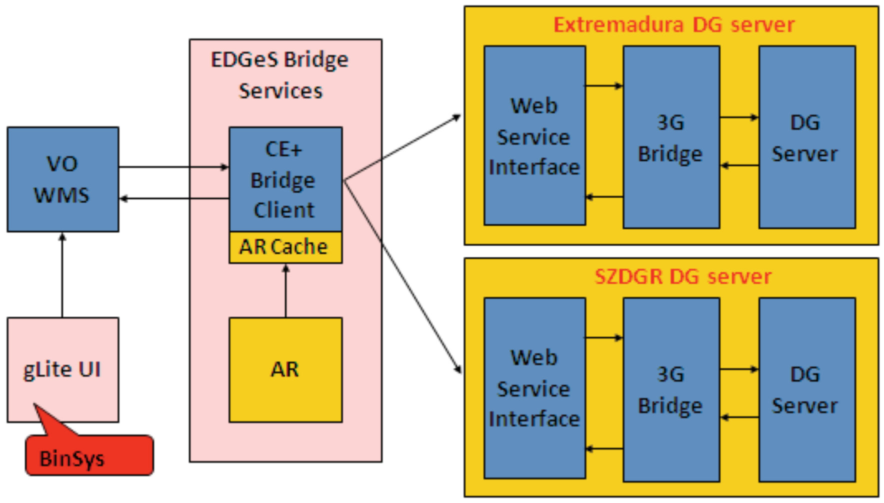

Figure 1.: EDGeS infrastructure for executing BinSys

Figure 1 shows the current DG infrastructure used by BinSys within EDGeS. Currently two Desktop Grids support executing BinSys applications. The first is the Red Tecnologica Eucativa DG[1] ("Extremadura DG") operated by Fundecyt in Spain, which has ~15000 hosts connected. The second is the SZTAKI Desktop Grid Research Facility[2] ("SZDGR DG"), which is used only for testing purposes, and has usually ~50 hosts connected from volunteers who donated their resources for testing applications. The

[1] http://edges-ext-dg.ceta-ciemat.es/rtedg/
[2] http://mishra.lpds.sztaki.hu/szdgr/

EDGeS Bridge services allow the submission of BinSys tasks from a gLite User Interface ("gLite UI") to a mixed EGEE-DG infrastructure, thus providing even more computational resources that the two desktop grids by themselves alone. The original DC-API version of BinSys can be used for gLite based execution in EGEE.

Results and conclusions

The execution of the application on the SZTAKI Desktop Grid Project [3] already provided the complete list of all binary (determinant 2) canonical number systems (CNS) up to degree 13.

c_0 \ Degree	2	3	4	5	6	7	8
2	5/4	7/4	29/12	29/7	105/25	95/12	309/32
3	7/5	25/13	131/47	310/75	1413/242	2619/332	10273/816
4	9/6	51/26	327/108	1240/286	6749/1033	20129/2194	
5	11/7	85/43	655/200	3369/735	21671/3010		
6	13/8	127/63	1155/332	7468/1546	55785/7106		
7	15/9	177/88	1829/509	14411/2876	122633/14606		
8	17/10	235/115	2747/742	25265/4887	241391/27263		
9	19/11	301/147	3905/1025	41331/7802			
10	21/12	375/182	5379/1378	63959/11824			

Table 1: The number of expansive and CNS polynomials, ordered by degree and constant term. (The missing cells are being computed.)

$-c_0$ / Degree	2	3	4	5	6	7	8
2	1/1	7/1	7/1	29/1	23/1	95/1	57/1
3	3/2	25/3	55/4	310/5	563/6	2619/7	4091/8
4	5/3	51/6	179/10	1240/15	3605/21	20129/28	
5	7/4	85/10	421/20	3369/35	13501/56		
6	9/5	127/15	795/35	7468/70	37853/126		
7	11/6	177/21	1353/56	14411/126	88501/252		
8	13/7	235/28	2099/84	25265/210	182235/462		
9	15/8	301/36	3083/120	41331/330			
10	17/9	375/45	4349/165	63959/495			

Table 2: The number of expansive and semi CNS-polynomials, ordered by degree and constant term. (The missing cells are being computed.)

The number of CNS polynomials (resp. expansive polynomials) is 12 (192) for degree 9, 42 (623) for degree 10, 11 (339) for degree 11, 66 (1085) for degree 12 and 15 (526) for degree 13. The numbers up to degree 8 were known earlier.

The search also inspired the introduction of semi-GNS, of which the decimal number system is a special case. This concept handles the problem that negative numbers are not representable. On Table 1 and 2 we list the number of CNS and expansive polynomials as well as the number of semi-CNS and expansive polynomials for the known cases. The more special cases are covered, the better predictions we can make about the number and structure of systems with sizes currently unreachable by computational analysis.

We hope by executing BinSys on the EDGeS infrastructure, which accumulates vastly more resources than SZTAKI Desktop Grid, to provide results beyond degree 13.

Acknowledgments

The work presented in this paper was supported by the EDGeS FP7 project funded by the European Commission under contract number INFSO-RI-211727 and by the project TÁMOP-4.2.B-09/1/KMR of Eötvös Loránd University.

References

[1] Burcsi, P., Kovács, A.: Exhaustive search methods for CNS polynomials, Monatshefte für Mathematik, 155, 421-430, 2008

[2] Urbah, E., Kacsuk, P., Farkas, Z., Fedak, G., Kecskeméti, G., Lodygensky, O., Marosi, A. Cs., Balaton, Z., Caillat, G., Gombás, G., Kornafeld, A., Kovács, J., He. H., Lovas., R.: EDGeS: Bridging EGEE to BOINC and XtremWeb. Journal of Grid Computing, 7/3. 335-354. 2009.

[3] SZTAKI Desktop Grid Project. http://szdg.lpds.sztaki.hu/szdg/